GENESYS™

This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Platform SDK Deployment Guide

Planning Your Platform SDK Deployment

12/18/2025

Contents

Planning Your Platform SDK Deployment

Introduction

Platform SDK allows you to write .NET and Java applications that communicate with Genesys servers in their native protocols. You can think of the APIs in Platform SDK as "Server APIs," since each one unlocks the capabilities of the server it connects to. In contrast to the abstraction found in other Genesys SDKs, Platform SDK was designed to offer low-level components and fine-grained, message-driven interfaces which are also XML friendly.

Every Genesys product also includes a Release Note that provides any late-breaking product information that can often be important. Direct links to the latest Release Notes for this product are provided under the Release Information section of the product page.

What You Should Know

This document is primarily intended for application developers who are familiar with Java or .NET technologies and who are planning to develop customer applications for the Genesys Framework environment.

It has been written with the assumption that you have a basic understanding of:

- The underlying concepts and terminology for the type of application you plan to develop. For instance, an understanding of CTI technology is important for developing an application with Voice Platform SDK
- Network design and operation
- Your own network configurations

You should also be familiar with messaging-compliant programming, Java- and .NET-related development tools, and how client and server applications work.

Deployment Prerequisites

To work with Genesys Platform SDK, you must ensure that your system meets the requirements established by the Genesys System-Level Guides—in particular the *Genesys Supported Operating Environment Reference Guide* and the *Genesys Interoperability Guide*.

- Supported Operating Environment: Platform SDK
- 8.5 Interoperability with Configuration Layer Environment

In addition to system-level requirements, the following prerequisites must be met:

.NET Prerequisites

- All underlying servers which your custom applications will connect to must be release 7.2 or higher.
- .NET Framework 3.5 or later is installed on the computer where you plan to run your Platform SDK application. (The .NET Framework can be downloaded free of charge from Microsoft).
- Starting with release 9.0.001.02, the NLog third-party library is required to use the Platform SDK NLog Adapter and to build the Application Template Application Block for .Net.

3rd party Component Name	License	Download URL
NLog 4.4.x	BSD license	NLog

Java Prerequisites

- All underlying servers which your custom applications will connect to must be release 7.2 or higher.
- Refer to the Genesys Supported Operating Environment Reference Manual for supported versions of JDK. Note that application blocks may require a different version of JDK.
- For Linux installations, the Linux compatibility packages must be installed prior to installing the Genesys IPs.
- To use the Platform SDK Jackson JSON Module (`jackson2-module.jar`), you must first download the following third-party component libraries:

3rd party Component Name	License	Download URL
Jackson Core 2.x	APL2	Jackson Download
Jackson Annotations 2.x		
Jackson Databind 2.x		

Platform SDK and AES Cryptography

Platform SDK uses AES cryptography as part of the Configuration Platform SDK API. AES encryption uses a 128-bit encryption key, which is considered too strong for some countries and thus subject to export restrictions. Please check the documentation provided for your Java environment for more details.

- http://docs.oracle.com/javase/8/docs/technotes/guides/security/overview/jsoverview.html
- http://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html

Support for Java 9 or higher

If working with Java 9 or higher, there are some special considerations you need to be aware of when building your Platform SDK application.

Platform SDK Configuration Protocol

When Platform SDK applications that use the Configuration Protocol are running with Java 9 or higher, the application may experience runtime exceptions such as "java.lang.NoClassDefFoundError: javax/xml/bind/JAXBException". This happens because JAXB APIs are considered to be Java EE APIs, and therefore are no longer contained in the default class path for Java SE 9. In Java 11 and subsequent releases, these APIs are completely removed from the JDK.

To resolve this issue, the Platform SDK POM files have been updated to contain required dependencies for this functionality starting in release 9.0.004.04. If your application was built using earlier releases of Platform SDK, or if your application was started without classpath construction coming from the Platform SDK POM files, you can resolve this issue by adding user application dependencies, as shown below:

```
<!-- import bom-->
<dependencyManagement>
   <dependencies>
      <dependency>
         <groupId>org.glassfish.jaxb</groupId>
         <artifactId>jaxb-bom</artifactId>
         <version>2.3.1</version>
         <type>pom</type>
         <scope>import</scope>
      </dependency>
   </dependencies>
</dependencyManagement>

<!-- required dependencies from bom-->
<dependencies>
  <dependency>
     <groupId>org.glassfish.jaxb</groupId>
     <artifactId>jaxb-runtime</artifactId>
  </dependency>
  <dependency>
     <groupId>javax.xml.bind</groupId>
     <artifactId>jaxb-api</artifactId>
  </dependency>
</dependencies>
```

A more detailed explanation, along with suggestions for alternative solutions, can be found from the following articles:

- http://openjdk.java.net/jeps/320
- https://stackoverflow.com/questions/43574426/how-to-resolve-java-lang-noclassdeffounderror-javax-xml-bind-jaxbexception-in-j

Compiling Platform SDK and the AppTemplate Application Block, or Using Application-Specific LMS files with the AppTemplate Generator

Platform SDK uses common java annotations, which were moved out of Java SE (as a part of Java EE). When you try to compile Platform SDK and its Application Blocks, or try to use the AppTemplate LMS enumeration generation tool for application-specific LMS files support, a compilation time exception like following may occur:

```
[ERROR] ...java:[19,24] package javax.annotation does not exist
[ERROR] ...java:[23,2] cannot find symbol
```

```
[ERROR] symbol: class Generated
```

To resolve this issue, you can add user application dependencies, as shown below:

```
<dependency>
    <groupId>javax.annotation</groupId>
    <artifactId>javax.annotation-api</artifactId>
    <version>1.3</version>
</dependency>
```